KATHOLIEKE UNIVERSITEIT
LEUVEN

Faculty of Business and Economics

Self-imposed time windows in vehicle routing problems

O. Jabali, R. Leus, T. Van Woensel and A.G. de Kok

DEPARTMENT OF DECISION SCIENCES AND INFORMATION MANAGEMENT (KBI)

KBI 1028

Self-Imposed Time Windows in Vehicle Routing Problems

October 21, 2010

Abstract

We speak of Self-Imposed Time Windows (SITW) when a logistics service provider quotes a delivery time window to his customer. Once this time window is communicated, the company strives to respect it as well as possible. We incorporate these SITW within the framework of the Vehicle Routing Problem (VRP). Essential to SITW is the fact that the time window is determined by the carrier company and not by the customer. The resulting VRP-SITW is inherently different from the well-studied VRP with Time Windows (VRPTW) in that in the latter problem the time windows are exogenous constraints imposed by the customers. The second important element of the problem studied in this paper is the uncertainty in the travel times. The basic mechanism of dealing with this uncertainty is the allocation of time buffers throughout the routes, which absorb disruptions. We propose a heuristic solution approach combining an LP model and a local search heuristic. A tabu search heuristic assigns customers to vehicles and establishes the order of visit of the customers per vehicle. Detailed timing decisions are subsequently generated by the LP model, whose output also guides the local search in a feedback loop. We test our algorithm on a number of benchmark instances for the VRP and VRPTW. We highlight the costs involved in integrating SITW with the VRP and we underline the advantages to the carrier company of SITW when compared to VRPTW.

Keywords: routing; vehicle routing problem; vehicle scheduling; disruptions in travel times; tabu search; linear programming.

1. Introduction

Many small-package shipping companies provide their customers with a time window for delivery and display this in their online tracking system. UPS, for instance, shows information on the delivery time window of orders for DELL computers. Obviously, once a time window is quoted to the customer, the carrier company wants to service the client within this window and so this should be reflected in the carrier's routing decisions. The described environment is clearly distinct from both the classic Vehicle Routing Problem (VRP) as well as from the VRP with Time Windows (VRPTW). It is different from the VRP since the VRP objective is to minimize the operational costs (e. g., distances or travel times (Laporte, 1992)). The VRPTW, on the other hand, does consider time windows but assumes they are exogenous, i. e., imposed by the customer (Bräysy and Gendreau, 2005a,b). As a consequence, the VRPTW imposes restrictions on the specific arrival times at each customer, while maintaining the objective of minimizing operational costs.

Our problem at hand considers time windows but treats them as endogenous to the VRP model. Specifically, the carrier company assigns customers to vehicles, sequences the customers allocated to each vehicle, and sets the time windows in which it plans to serve the customers. In the remainder of this paper, we will refer to the described problem as the Vehicle Routing Problem with Self-Imposed Time Windows (VRP-SITW). The term 'self-imposed' refers to the fact that the carrier company selects the time windows by itself, independently of the customer. Once the time windows are quoted to the customer, however, the customer should be serviced within the window. As such, the VRP-SITW conceptually lies in between the VRP and the VRPTW. We assume that service cannot start before the time window, leading to waiting in case of early arrivals. Furthermore, late arrivals are permitted but penalized proportionally to their tardiness. Drivers have a fixed shift length and are paid a fixed amount per day. Finally, disruptions in traveling time may occur between each two customers. This mainly reflects accidents, weather condition, vehicle breakdown or road works. One natural way to protect schedules against this uncertainty is to include time buffers (see, for instance, Hopp and Spearman (1996) for a similar logic in a production environment). Inspired by the scheduling literature, we propose a buffer allocation model that inserts slack time into the schedule to cope with possible delays. Our solution framework relies on the tabu search heuristic for assigning customers to routes and for the sequencing of each route. The actual evaluation of the target function is achieved by solving the resulting buffer allocation model to optimality for each route separately; this sub-problem is a linear programming problem. In the terminology of Puchinger and Raidl (2005), our hybrid algorithm is collaborative, since there is a clear hierarchy between the two phases. Examples of earlier works that combine local search with LP are Finke et al. (2007), where job-machine allocation is performed via tabu search while an LP model is used for inserting buffers in between jobs. Flisberg et al. (2009) solve a VRPTW via tabu search based on the input of an LP that defines origins and destinations for full truckloads.

The parallelism between vehicle routing and production scheduling is highlighted by Gendreau et al. (1995), who study single-vehicle routing and scheduling to minimize the number of delays. Given a deadline for servicing each customer, the objective is to minimize the number of late deliveries. The problem is equivalent to single-machine scheduling with sequence-dependent setup times to minimize the number of tardy jobs. The scheduling aspect is fundamental in Mitrović-Minć and Laporte (2004), in the context of dynamic pickup and delivery with time windows. The authors first solve the routing component and then look into the scheduling component. Four waiting strategies are presented and assessed based on the distance along with the number of vehicles required. Xiang et al. (2008) study the dynamic dial-a-ride under various types of uncertainty. They propose several scheduling strategies for handling dynamic events, accounting for a fixed duration and overtime costs in the case of exceeding the shift length. Our problem VRP-SITW differs from the above literature in that customer demand is known in advance. Stochastic travel times in VRP are investigated in Laporte et al. (1992), where vehicles incur penalties for exceeding a limit on the route duration. Li et al. (2010) examine VRPTW with stochastic travel and service times. Their model also includes overtime costs for exceeding route duration and soft time windows; the actual penalties are computed by means of simulation.

The main contributions of this paper are threefold: (1) we describe the new yet practical setting of SITW in vehicle routing; (2) we describe how a VRP with SITW and stochastic travel times can benefit from time buffers; and (3) we develop a hybrid LP / tabu search algorithm for producing high-quality solutions. Our aim is to construct a stable *a priori* plan that best copes with disruptions; in other words, a solution is generated at the start of the planning horizon and does not require further optimization during its implementation.

The remainder of the paper is organized as follows. We provide a number of definitions and a detailed problem statement in Section 2. Our solution procedure is described in Section 3. The computational experiments are presented and discussed in Section 4. Finally, in Section 5, we highlight the main results and indicate directions for future research.

2. Description of VRP-SITW

Consider a set of N customers with a fleet of K identical vehicles. Each customer i has a demand q_i and is to be serviced by a single vehicle. The logistics network is represented by a complete directed graph $G = (V, A)$, with $V = \{0, 1, \ldots, N\}$ the set of vertices and A the set of directed links. The vertex 0 denotes the depot; the other vertices of V represent the customers. The non-negative weight d_{ij} associated with each arc (i, j) represents the distance from i to j. Each vehicle must start and end its route at the depot, the total demand on each route cannot exceed the vehicle capacity Q and each customer should be visited exactly once. The objective of the VRP is to construct routes that bring the total travel time of the vehicles to a minimum. The VRP-SITW entails the same elements as the VRP but with a number of additional parameters. Below, we first give a general description of the objective function (Section 2.1). Subsequently, we provide we elaborate on the SITW model and on the way in which stochasticity is captured (in Sections 2.2 and 2.3, respectively).

2.1 Objective function

The objective function of the VRP-SITW consists of three parts. The first part is the travel cost, which captures the vehicle operating costs such as fuel costs. The second part of the objective function is a tardiness penalty, which represents the desire to respect the quoted time windows as well as possible. A 'railroad-scheduling approach' is adopted: the lower bound of the time window is the earliest starting time of the service. Arrival before the scheduled window is not penalized, since the driver cost is presumed to be fixed. Arrival after the time window, however, leads to a penalty proportional to the tardiness. The third component of the objective function is an overtime penalty. We suppose that the drivers are paid a fixed amount for a shift with fixed duration; if this duration is exceeded then overtime penalties are due.

In the optimal solution to the VRP-SITW, the travel time will never be less than for the associated VRP instance since the latter has neither tardiness nor overtime penalties. The travel time in optimal solutions to VRP-SITW and VRPTW is in principle incomparable, since the fixed time windows are relaxed in the former but there are extra penalties in the objective. The computational experiments described in Section 4 indicate that the VRP-SITW leads to less travel time in most of the instances studied, presumably because the time windows are decision variables rather than constraints. With travel costs only, the VRP-SITW is equivalent to the VRP and is thus NP-hard.

A solution to the VRP-SITW is a set of routes $Z = \{R_1, R_2, \ldots, R_{|Z|}\}$ with $|Z| \leq K$. Each route R_r ($r = 1, \ldots, |Z|$) is a vector $(0, i, j, \ldots, 0)$ whose components are elements of V, specifying which clients (vertices) will be visited by the vehicle following the route, and in which order. Each route begins and ends at the depot (vertex 0) and each vertex different from 0 belongs to exactly one route. We say that $i \in R_r$ if the vertex $i \in V$ is part of route $R_r \in Z$ and $(i, j) \in R_r$ if i and j are two consecutive vertices in R_r. The objective function for the VRP-SITW is then

$$F(Z) = \Omega(Z) + \sum_{R_r \in Z} \Theta(R_r), \tag{1}$$

with $\Omega(Z)$ the total travel cost associated with solution Z and $\Theta(R_r)$ representing the overtime and tardiness penalties of route R_r. The travel cost is defined as follows:

$$\Omega(Z) = c \sum_{R_r \in Z} \sum_{(i,j) \in R_r} d_{ij},$$

with c the cost of traveling one unit of distance. The penalties of each route are evaluated by solving a buffer allocation problem, which is described in Section 2.2.

2.2 Self-imposed time windows

Each route R_r consists of a set of n_r customers. For convenience, when referring to one specific route, we relabel the customers in ascending order: $R_r = (0, 1, 2, \ldots, n_r, n_r + 1)$, where the depot corresponds with $0 \equiv n_r + 1$. The distance $d_{i,i+1}$ between consecutive nodes i and $i+1$ in the route is written as d_i. A *schedule* for route R_r is an $(n_r + 2)$-vector $\mathbf{s} = (s_0, s_1, \ldots, s_{n_r+1})$, specifying a departure time s_i from each node $i \in R_r$. The shift length is the time interval $[s_s, s_e]$, implying that $s_s \leq s_0$. Each customer $i \in R_r \setminus \{0, n_r + 1\}$ has a time-window length W_i within which the arrival of the vehicle is desired. The carrier company communicates time windows to its customers based on the schedule $\mathbf{s}$. Each node $i \in R_r$ also has a standard service time u_i, e. g., for load/unload activities. We assume that a vehicle will never leave a customer earlier than scheduled. The left bound of the time window is then $s_i - u_i$, as this constitutes an earliest starting time for the servicing operations. An illustration is provided in Figure 1. The service times u_0 and u_{n_r+1} at the depot are set to zero.

During the realization of this baseline schedule, disruptions might occur. We examine disruptions corresponding with an increase in the travel time d_i between customers i and $i + 1$. The length L_i of this delay is a random variable, which is modeled by means of discrete scenarios; a similar choice in a machine-scheduling context is made by, e. g., Daniels and Carrillo (1997), Daniels and Kouvelis (1995), Kouvelis et al. (2000), and Kouvelis and Yu (1997). Specifically, we let L_i denote the increase in d_i if i is 'disrupted', which takes place with probability p_i. The variable L_i is discrete with probability-mass function $g_i(\cdot)$, which associates non-zero probability with positive values $l_{ik} \in \Psi_i$, where Ψ_i denotes the set of disruption scenarios for d_i, so $\sum_{k \in \Psi_i} g_i(l_{ik}) = 1$. We use g_{ik} as shorthand for $g_i(l_{ik})$; the disruption lengths l_{ik} are indexed from small to large for a given i. The realization of L_i becomes known only when arc $(i, i+1)$ is traversed. The actual departure time at customer i is denoted by $s_i^a(\mathbf{s})$; this is a random variable that is dependent on the schedule $\mathbf{s}$ (in the remainder of the article, we omit the argument $\mathbf{s}$ when there is no danger of confusion). The value $s_i - u_i$ is a lower bound on the starting time of the client's service. This so-called railroad-scheduling approach implies that $s_i \leq s_i^a$, $\forall i \in R_r$, and guarantees that the actual schedule will strictly copy the baseline schedule if no disruptions occur. In effect, the scheduled times become 'release dates' for departure times s_i^a from each customer $i \in R_r$:

$$
\begin{aligned}
s_0^a &= s_0 \\
s_i^a &= \max\{s_i; s_{i-1}^a + d_{i-1} + L_{i-1} + u_i\}, \qquad i = 1, \ldots, n_r + 1.
\end{aligned}
$$

Arrival prior to $s_i - u_i$ is not penalized. With arrival later than $s_i - u_i + W_i$, however, we associate a cost proportional to the tardiness: a non-negative integer penalty t_i is incurred

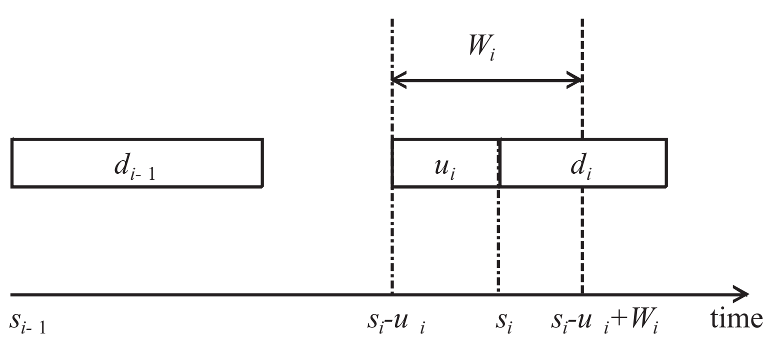

Figure 1: Illustration of a time window at customer i

per unit-time delay. The value t_{n_r+1} is the cost for arriving late at the depot at the end of the tour.

We assume that the driver receives a fixed payment for the shift, which ends at s_e. Arrival after the end of the shift incurs an overtime penalty b per time unit. We can now elaborate the penalty term $\Theta(\cdot)$ in Equation (1). For a given route R_r, $\Theta(R_r)$ consists of two components, namely the expected delay costs at customers and at the depot on the one hand, and the expected overtime penalty on the other hand. Specifically,

$$\Theta(R_r) = \sum_{i \in R_r \setminus \{0\}} t_i \mathbb{E}[\max\{0; s_i^a(\mathbf{s}) - (s_i - u_i + W_i)\}] + b\mathbb{E}[\max\{0; s_{n_r+1}^a - s_e\}], \qquad (2)$$

with $\mathbb{E}[\cdot]$ the expectation operator (note that $\mathbf{s}$ is actually also a parameter to $\Theta(\cdot)$). In the following subsection, we outline the disruption model in detail.

2.3 Modeling disruptions

When the durations are independent, little less is possible for objective-function evaluation than to consider all $\prod_{i \in R_r \setminus \{n_r+1\}} (|\Psi_i| + 1)$ possible combinations of duration disruptions. This was the motivation in a scheduling context in Herroelen and Leus (2004); Leus and Herroelen (2005, 2007); Ballestín and Leus (2008) to develop a model that considers only the main effects of the separate disruption of each of the individual jobs rather than all possible disruption interactions. Computational results in the aforementioned scheduling applications show that the resulting model is quite robust to variations in the actual number of disrupted jobs. In the context of time-dependent VRP with service disruptions, Jabali et al. (2009) also focus on the effects of single disruptions. We make a similar assumption in this paper: our model assumes that exactly one leg will suffer a disruption from its baseline duration. The underlying practical motivation is that we should only optimize for one 'inconvenience' per day, as it would be very difficult to protect from multiple disruptions at multiple places at multiple times. The resulting restricted model is useful when disruptions are sparse and spread over time so that the number of interactions is limited.

For a given route R_r we distinguish between two situations: either no leg in R_r is disturbed, or a single leg is disturbed in R_r. Let ζ denote the overtime for R_r when no leg is disturbed (tardiness penalties are irrelevant if no leg is disturbed). The total penalty $\Theta(R_r)$ consists of two components, namely the expected delay costs at customers and at the depot on the one hand, and the expected overtime penalty on the other hand. Specifically, for a given route R_r, under the one-disruption assumption and with $s_{i-1} + d_{i-1} + u_i \leq s_i$ for all $i > 0$, the relevant penalty term in (2) can be written as

$$\Theta(R_r) = \sum_{i=0}^{n_r} \sum_{j=i+1}^{n_r+1} \sum_{k=1}^{|\Psi_i|} p_i g_{ik} t_j \Delta_{ijk} + b \sum_{i=0}^{n_r} \sum_{k=1}^{|\Psi_i|} p_i g_{ik} \Lambda_{ik} + b\big(1 - \sum_{i=0}^{n_r} p_i\big)\zeta.$$

In this expression,

$$\Delta_{ijk} = \max\left\{0 \quad ; \quad s_i + d_i + l_{ik} + \sum_{m=i+1}^{j-1} (u_m + d_m) - s_j + u_j - W_j\right\},$$
$$i \in R_r \setminus \{n_r + 1\}; j \in R_r \setminus \{0\}; i < j; k \in \Psi_i,$$

$$\Lambda_{ik} = \max\{0 \quad ; \quad s_{n_r+1} + \Delta_{i,n_r+1,k} - s_e\}, \qquad i \in R_r \setminus \{n_r + 1\}; k \in \Psi_i$$

and

$$\zeta = \max\{0 \quad ; \quad s_{n_r+1} - s_e\}.$$

Remember that p_i represents the probability that d_i is the unique disrupted value. The variable Δ_{ijk} represents the tardiness at client j due to a disruption according to scenario k of d_i, which is equal to zero or to the disruption length of i minus the buffer size in place between the customers i and j, whichever is larger. The term $\sum_{m=i+1}^{j-1} (u_m + d_m)$ is the service and travel time for the customers between i and j. Similarly, Λ_{ik} is the overtime resulting from a disruption at customer i by scenario k. The overtime is zero in case of arrival at the depot before the shift end s_e, and equal to the realized arrival time minus s_e

otherwise. Thus, ζ is zero in case of arrival at the depot before the shift end. The probability that a route is not disturbed is $(1 - \sum_{i=0}^{n_r} p_i)$.

3. A hybrid solution procedure

Our solution method for the VRP-SITW proceeds in two stages: first routing and then scheduling. The assignment of customers to vehicles and the sequencing of customers are done in stage 1; this stage uses tabu search. Iteratively, the routes generated by the tabu search are then scheduled in the second stage, where we use linear programming to solve the sub-problem to optimality under the one-disruption assumption. We say that our solution procedure is 'hybrid' due to the combined use of a meta-heuristic and an exact optimization routine. Below, we first describe the lower-level scheduling problem in Section 3.1, followed by the tabu search procedure (Section 3.2).

3.1 Scheduling and buffer insertion

For a given route R_r, the linear program below produces an optimal schedule, conditional on exactly one leg being disrupted. Buffer sizes are implicit from the resulting schedule.

$$\Theta(R_r) = \min \qquad \sum_{i=0}^{n_r} \sum_{j=i+1}^{n_r+1} \sum_{k=1}^{|\Psi_i|} p_i g_{ik} t_j \Delta_{ijk} + b \sum_{i=0}^{n_r} \sum_{k=1}^{|\Psi_i|} p_i g_{ik} \Lambda_{ik} + b\big(1 - \sum_{i=0}^{n_r} p_i\big)\zeta$$

subject to

$$
\begin{aligned}
s_{i-1} + d_{i-1} + u_i &\leq s_i \qquad\qquad i \in R_r \setminus \{0\} && (3)\\
s_0 &\geq s_s && (4)\\
s_i + d_i + l_{ik} + \sum_{m=i+1}^{j-1} (u_m + d_m) &\leq s_j - u_j + W_j + \Delta_{ijk} && \\
& \qquad i \in R_r \setminus \{n_r+1\}; j \in R_r \setminus \{0\}; i < j; k \in \Psi_i && (5)\\
s_{n_r+1} + \Delta_{i,n_r+1,k} - s_e &\leq \Lambda_{ik} \qquad\qquad i \in R_r \setminus \{n_r+1\}; k \in \Psi_i && (6)\\
\zeta &\geq s_{n_r+1} - s_e && (7)\\
\text{all } \Delta_{ijk} &\geq 0;\ \text{all } s_i \geq 0;\ \text{all } \Lambda_{ik} \geq 0;\ \zeta \geq 0 && (8)
\end{aligned}
$$

Constraints (3) can be viewed as precedence constraints: the scheduled departure time s_i from customer i is at least equal to the departure time of its predecessor s_{i-1} augmented with the distance d_{i-1} and the service time u_i. This implies that the buffer between customers $i-1$ and i is $s_i - (s_{i-1} + d_{i-1} + u_i)$. Constraint (4) ensures that the scheduled departure time from the depot does not precede the shift's start time s_s. Constraints (5), (6) and (7) determine the delay terms Δ_{ijk}, Λ_{ik} and ζ, respectively, as described in Section 2.3.

3.2 Tabu search for the VRP-SITW

Tabu search has been widely used for solving the VRP, see for example Gendreau et al. (1994, 1996); Hertz et al. (2000). Furthermore, it has been extensively used to solve VRPTW as well, examples can be found in Garcia et al. (1994); Taillard et al. (1997). Thus, adopting the tabu search heuristic comes as a natural choice also for the VRP-SITW. Our tabu search procedure generates a set of routes that still need to be scheduled using the lower-level LP described in Section 3.1. The procedure iteratively scans the members of a neighborhood of the current solution to evaluate possible improvements in the objective function. Due to our bi-level approach, the evaluation of each neighborhood solution requires a separate

LP run, which, if performed to optimality, would require enormous computation times. We have therefore opted for approximations of these optimal overtime and tardiness penalties to guide the tabu search in selecting the best move in its current neighborhood. Once a move is selected, its exact target function is computed by invoking the LP model for the changed route or routes, leading to a new optimal schedule.

The overall procedure is described in pseudo-code as Algorithm 1. We adopt three different criteria C_1, C_2 and C_3 for choosing a move; these will be described in detail below. The tabu search procedure is run consecutively with each of the three criteria. The initial solution Z_0 is the output of the nearest neighbor heuristic for each of the three criteria. Feeding the best-found solution of C_1 into the run for C_2 and for C_2 into C_3 has been tested, together with many variations of the order of the three criteria, but this did not lead to better results. For each customer $i \in V$, we construct 2-opt* (Potvin and Rousseau, 1995) and Or-opt (Or, 1976) neighborhoods for the η nodes closest to i. A chosen move is declared tabu for the next κ iterations. The process iterates until a maximum number of non-improving moves is reached.

In line with Gendreau et al. (1994), diversification of the search is achieved by allowing demand-infeasible solutions (i. e., routes with total demand exceeding the vehicle capacity). Such infeasible solutions are penalized in proportion to their capacity violation by means of the following composite objective function, which replaces $\Omega(Z)$:

$$\Omega_2(Z) = \Omega(Z) + w \sum_{R_r \in Z} \left[\Big(\sum_{i \in R_r} q_i \Big) - Q \right]^+ . \tag{9}$$

In Equation (9) each unit of excess demand is penalized by a factor w. This excess penalty w is decreased by multiplication with a factor ν after ϕ consecutive feasible moves. Similarly, w is increased (multiplied by factor ν^{-1}) after ϕ infeasible iterations.

Below, we describe the three criteria that allow avoiding the use of the LP model for each candidate solution and lead to computationally efficient move selection procedures.

Algorithm 1 Global algorithmic structure

1: construct initial solution Z_0 and compute $F(Z_0)$
2: **for** $\xi = 1$ to 3 **do**
3: set $Z = Z_0$ and $F(Z) = F(Z_0)$
4: generate the neighborhood of Z
5: evaluate all neighbors on criterion C_ξ and retain the best non-tabu move as new solution Z
6: evaluate $F(Z)$ and update the tabu list to include Z
7: **if** Z is feasible and is better than the current best solution **then**
8: update the best feasible solution for C_ξ to Z
9: **end if**
10: update excess demand penalty
11: **if** no improvement in $\eta_{\max}$ iterations **then**
12: store best solution for C_ξ
13: **else**
14: go to step 4
15: **end if**
16: **end for**
17: return the best solution from $\xi = 1, 2$ and 3

C_1 **- distance based** This heuristic is based purely on minimizing the modified travel costs $\Omega_2(\cdot)$, i. e., it does not take into account the time windows and their associated penalties, nor does it consider overtime. Thus, C_1 is similar to the criteria used in local search for the VRP. Let Z' be a neighbor of the current solution Z and define $\Delta_1(Z') = \Omega_2(Z) - \Omega_2(Z')$. The chosen move is one that is not tabu and maximizes $\Delta_1(\cdot)$.

C_2 **- distance based and marginal penalties** This measure adds to C_1 an assessment of the penalty component $\sum_{R_r \in Z} \Theta(R_r)$. For given Z, the marginal penalty of route R_r is $\frac{\Theta(R_r)}{n_r+1}$. Consider a move involving two routes R_1 and R_2, leading to solution Z' . Let n_1 and n_2 be the number of nodes visited by routes R_1 and R_2, respectively, in the current solution Z, and n'_1 and n'_2 the number of nodes visited by routes R_1 and R_2 in the new solution Z'. C_2 picks the move that is not tabu and maximizes the following expression:

$$\Delta_2(Z') = \Omega_2(Z) - \Omega_2(Z') + \rho\Big[\Theta(R_1) + \Theta(R_2) - \frac{\Theta(R_1)}{n_1+1}(n_1'+1) - \frac{\Theta(R_2)}{n_2+1}(n_2'+1)\Big].$$

The logic behind this evaluation is based on the observation that penalties increase with the number of customers in the route. Decreasing the number of customers in a route with a large penalty value is likely to decrease the total objective value associated with the route.

C_3 **- distance and buffer based** As mentioned in Section 3.1, the buffer size between customers i and $i+1$ is $b(i) = s_{i+1} - (s_i + d_i + u_{i+1})$. Criterion C_3 favors moves with small buffers. Each buffer unit is penalized by γ. For each candidate solution Z' involving a move between customer i and customer j, we compute the following quantity:

$$\Delta_3(Z') = \Omega_2(Z) - \Omega_2(Z') - \gamma[b(i) + b(j)].$$

We chose a move that is not tabu and that maximizes $\Delta_3(\cdot)$. The reasoning involved in this move selection process is the following: improvements in travel times are more likely to also decrease the penalties when the buffers are small.

Different aspects of the problem are tackled by each criterion. The impact of a move on the travel time $\Omega_2(Z)$ is efficiently computed. The accurate impact of a move on the penalty component $\sum_{R_r \in Z} \Theta(R_r)$ of the target function, on the other hand, requires evaluation of the SITW model for the affected route or routes. Criteria 2 and 3 attempt to assess moves based on the penalty values of the current solution rather than via the LP model. We note that C_2 is equivalent to C_1 for moves involving a single route, which can occur only with Or-opt moves.

4. Computational experiments

We have run a number of experiments to assess the computational performance of our algorithm and to compare the outcomes of the VRP-SITW with both the results of the VRP and of the VRPTW. Throughout this section, the travel cost c in $\Omega(Z)$ is set to one, thus we use the terms distance and travel time interchangeably. For an instance with N nodes, for each customer the $\eta = \lceil 0.3N \rceil$ closest customers are candidates for a move. The tenure size κ is set to 20. The infeasibility penalty w equals 12, with $\phi = 5$ and $\nu = \frac{3}{4}$. The penalties associated with C_2 and C_3 are chosen as $\rho = 1$ and $\gamma = 0.1$, respectively. The overtime penalty b takes the value 2. The probability p_i is set to one over the total number of legs in a solution. Given a solution with k vehicles, where $k \leq K$, $p_i = \frac{1}{N+k}$. Hence, the probability of disruption is identical for all the legs in the solution.

We consider four disruption scenarios for each leg: $|\Psi_i| = 4$. The probabilities of disruption are also the same for each leg i, namely $g_{i1} = 0.5$, $g_{i2} = 0.3$, $g_{i3} = 0.1$ and $g_{i4} = 0.1$. Finally, the disruption lengths between customers i and j are assumed proportional to the baseline duration d_{ij}, namely $l_{i1} = 0.1d_{ij}$, $l_{i2} = 0.2d_{ij}$, $l_{i3} = 0.5d_{ij}$ and $l_{i3} = d_{ij}$.

All experiments are performed on a Intel(R)Core Duo with 2.40 GHz and 2 GB of RAM. The implementation is coded in C++. The LP instances are solved by embedding Gurobi Optimizer 2.0.2, which uses the simplex algorithm. The reported run times are in seconds. We have adopted two datasets from the literature. The first dataset contains a number of VRP instances from Augerat et al. (1998). We work with 27 VRP instances, with the number of customers ranging from 31 to 79. The vehicle capacity Q is 100 units. The baseline service time u_i for each customer i is set to 10 minutes. The shift start time and end time s_s and s_e are chosen as zero and 200, respectively. The window length W_i equals 60 for all i. The second dataset contains VRPTW instances and stems from Solomon (1987). We consider 29 instances with 100 customers (sets R1 (random), C1 (clustered) and RC1 (random and clustered)). The baseline service times u_i and window sizes W_i are given. The opening hours of the depot are used to determine the shift's starting time s_s and ending time s_e. The vehicle capacity Q is 200 units.

Below, we first conduct some experiments related to move selection and tardiness choices (in Section 4.1 and 4.2, respectively), followed by comparisons with VRP (Section 4.3) and with VRPTW (Section 4.4).

4.1 Move selection

Table 1 shows the results of implementations for the Augerat instances in which only one of the three criteria C_1, C_2 and C_3 is used during the optimization; the tardiness penalty $t_i = 5$ for all arcs. The left side of the table displays the target function value $F(Z)$ attained. The right side of the table exhibits the run time for each of the three measures. We observe that C_3 outperforms the other two critera in 15 out of the 27 instances, while C_1 and C_2 do so in seven and five instances, respectively. On average, C_1 requires less runtime than C_2 and C_3. The average run time over all heuristics is 17.3 minutes. Since we are working in an *a priori* setting, these running times are acceptable.

Table 2 contains similar results for the Solomon instances. The computation times are larger than those for the first dataset. This is partly due to a greater number of customers, but more importantly the number of customers per route is also larger than before. Thus, the LP subroutine will consume considerably more time. We note that we obtain identical results for some of the instances, which is due to the fact that the time window constraints in these VRPTW instances are now relaxed, and some of instances have the same time window lengths and customer locations. In line with Table 1, the three move selection criteria differ in performance. C_2 performs best in 23 out of the 27 instances, while this occurs for C_1 and C_3 in two and four instances, respectively.

4.2 Tardiness penalty choices

In order to evaluate the effects of varying delay penalty costs t_i, we have conducted experiments under four different cost settings, which are subsequently referred to as 'P5', 'P10',

Instance	Objective value			CPU time			
	C_1	C_2	C_3	C_1	C_2	C_3	Total
32 k5	955.4	1038.2	957.2	734	103	1568	2405
33 k5	744.8	724.1	716.6	78	121	166	365
33 k6	801.1	798.7	791.0	213	151	177	541
34 k5	867.9	876.1	852.7	335	135	374	844
36 k5	958.0	990.1	950.5	552	222	438	1212
37 k5	765.6	811.8	798.6	338	394	210	942
37 k6	1071.1	1069.0	1080.5	112	158	148	418
38 k5	822.6	832.5	823.4	361	299	227	887
39 k5	1013.1	957.5	995.9	200	302	289	791
39 k6	963.0	956.1	952.7	184	130	151	465
44 k6	1102.9	1057.7	1054.7	128	124	175	427
45 k6	1078.0	2685.8	1096.4	1117	71	1142	2330
45 k7	1294.4	1281.4	1302.8	80	86	82	248
46 k7	1072.5	1059.0	1008.7	99	221	401	721
48 k7	1256.3	1243.1	1247.2	169	230	224	623
53 k7	1185.3	1194.7	1165.3	192	1046	376	1614
54 k7	1293.7	1396.7	1335.5	253	446	320	1019
55 k9	1158.7	1137.4	1132.2	340	212	255	807
60 k9	1509.2	1489.4	1473.8	108	112	177	397
61 k9	1197.9	1239.7	1177.3	225	224	214	663
62 k8	1509.7	1516.0	1499.5	295	893	386	1574
63 k10	1556.2	1411.1	1493.0	157	607	292	1056
63 k9	1834.5	1897.8	1840.8	343	317	712	1372
64 k9	1658.5	1626.5	1587.8	202	431	521	1154
65 k9	1319.7	1307.3	1293.2	137	1249	115	1501
69 k9	1254.5	1276.8	1291.3	616	452	552	1620
80 k10	2095.0	2057.7	2046.5	399	1002	693	2094
Average				295	361	385	1040

Table 1: Comparison of the three move selection criteria for the Augerat instances

'Prob' and '1.3dist'. In P5, we choose $t_i = 5, \forall i \in V \setminus \{0\}$ (which was the choice also in Section 4.1), while P10 corresponds to $t_i = 10$. Under setting Prop, the delay cost for each customer equals the quantity ordered, so $t_i = q_i, \forall i \in V \setminus \{0\}$, which represents a situation where the delay penalty is proportional to the demand. The final experimental setting, denoted by 1.3dist, puts t_i equal to 5 for all customers, similarly to P5, but all distances are now increased by 30%. In this way, there is less slack time available, leading to less buffer time to be allocated and resulting in tighter instances.

Table 3 summarizes the results for the four experimental settings after running the full tabu search procedure (with the three criteria combined). The left side of the table shows the achieved target function values. Value $M(C_i)$ denotes the number of times (out of 27)

Instance	Objective value			CPU time			
	C_1	C_2	C_3	C_1	C_2	C_3	Total
R101	918.6	905.7	922.4	1773	3388	775	5936
R102	918.2	922.5	922.1	1724	1860	769	4353
R103	918.2	922.5	922.1	1734	1865	774	4373
R104	917.2	917.0	920.5	1737	3445	771	5953
R105	917.0	908.8	920.1	1722	2412	767	4901
R106	917.0	908.8	920.1	1743	2384	761	4888
R107	917.0	908.8	920.1	1752	2362	773	4887
R108	917.0	908.8	920.1	1765	2392	764	4921
R109	917.0	908.8	920.1	1728	2375	767	4870
R110	917.0	908.8	920.1	1730	2240	761	4731
R111	917.0	908.8	920.1	1717	2229	768	4714
R112	917.0	908.8	920.1	1743	2240	761	4744
C101	834.7	834.6	859.2	805	3209	1315	5329
C102	834.7	834.6	859.2	802	3181	1328	5311
C103	834.7	834.6	859.2	799	3210	1317	5326
C104	834.7	834.6	859.2	807	3271	1324	5402
C105	834.7	834.6	859.2	796	3308	1317	5421
C106	834.7	834.6	859.2	799	3296	1316	5411
C107	834.7	834.6	859.2	798	3211	1327	5336
C108	834.7	834.6	859.2	803	3187	1319	5309
C109	834.7	834.6	859.2	792	3198	1327	5317
RC101	1024.5	1013.2	1022.6	1198	1318	1071	3587
RC102	1024.5	1013.2	1022.6	1196	1318	1075	3589
RC103	1024.5	1013.4	1022.6	1195	1740	1121	4056
RC104	1024.5	1042.0	1022.6	1201	1026	1137	3364
RC105	1025.0	1013.6	1023.2	1195	1318	1093	3606
RC106	1024.5	1042.0	1022.6	1189	1024	1083	3296
RC107	1024.5	1042.0	1022.6	1209	1021	1090	3320
RC108	1024.5	1042.0	1022.6	1191	1023	1083	3297
Average				1298	2347	1029	4674

Table 2: Comparison of the three move selection criteria for the Solomon instances

that criterion C_i produces the best result; these values are presented in the last three lines of the table. Measure C_3 performs best in more instances in all four experimental settings. The best result for C_3 is in P5. In total, C_2 and C_3 perform best in 30 and 26 instances, respectively, when considering all four experimental settings. The fact that C_3 accounts for buffer sizes between customers might explain its superior performance.

On average, the objective values for P10 are only 0.7% higher than for P5. This means that even doubling the customer delay penalty does not affect the final objective value to a large extent. With varying penalties, as in the Prop setting, the values are not dramatically different either. For the case of 1.3dist, the average objective increase is 36.1% compared to

Instance	Objective				Penalty ratio			
	P5	P10	Prop	1.3dist	P5	P10	Prop	1.3dist
32 k5	955.4	961.7	956.6	1290.8	16.6%	17.1%	16.7%	18.5%
33 k5	716.6	716.9	716.5	998.7	6.3%	5.0%	6.3%	12.4%
33 k6	791.0	796.8	797.6	1066.9	5.0%	5.7%	5.8%	8.9%
34 k5	852.7	857.6	857.2	1190.6	7.0%	7.6%	7.6%	13.4%
36 k5	950.5	960.3	957.8	1285.6	13.5%	14.4%	14.2%	17.8%
37 k5	765.6	766.8	766.4	1101.0	10.9%	11.0%	11.0%	17.0%
37 k6	1069.0	1079.4	1079.3	1457.3	9.2%	9.6%	9.6%	13.3%
38 k5	822.6	824.3	824.0	1162.0	7.6%	7.8%	7.8%	15.2%
39 k5	957.5	971.6	969.4	1283.2	11.2%	11.6%	11.4%	15.3%
39 k6	952.7	957.8	953.4	1295.1	10.5%	11.0%	8.7%	16.0%
44 k6	1054.7	1059.6	1059.2	1489.6	8.8%	9.3%	9.2%	13.4%
45 k6	1078.0	1081.3	1066.1	1469.0	8.4%	8.6%	8.2%	12.7%
45 k7	1281.4	1298.3	1277.7	1713.5	7.7%	8.9%	8.5%	11.3%
46 k7	1008.7	1007.5	1009.4	1371.5	7.0%	6.9%	7.1%	10.6%
48 k7	1243.1	1244.1	1231.5	1662.6	10.2%	9.0%	8.2%	11.8%
53 k7	1165.3	1168.2	1167.1	1542.7	7.4%	7.6%	7.5%	11.7%
54 k7	1293.7	1302.2	1302.5	1799.6	7.6%	8.2%	8.2%	12.3%
55 k9	1132.2	1135.5	1136.8	1506.1	2.7%	2.9%	3.0%	5.1%
60 k9	1473.8	1482.7	1485.1	1980.8	5.5%	5.1%	6.2%	8.0%
61 k9	1177.3	1178.6	1178.5	1651.2	3.4%	3.6%	3.5%	6.6%
62 k8	1499.5	1505.7	1486.1	1986.3	9.4%	9.7%	8.7%	11.9%
63 k10	1411.1	1500.0	1501.2	1914.9	4.0%	4.1%	4.2%	6.8%
63 k9	1834.5	1847.7	1844.1	2472.9	8.5%	9.2%	9.0%	11.3%
64 k9	1587.8	1598.7	1597.1	2166.2	8.5%	9.1%	9.0%	11.3%
65 k9	1293.2	1295.3	1293.7	1720.5	3.0%	3.2%	3.1%	6.4%
69 k9	1254.5	1256.6	1256.7	1643.9	4.0%	4.1%	4.1%	6.4%
80 k10	2046.5	2061.4	2042.8	2756.5	9.9%	10.6%	9.1%	11.7%
Average penalty %					7.9%	8.2%	8.0%	11.7%
$M(C_1)$	7	8	7	8				
$M(C_2)$	5	5	7	9				
$M(C_3)$	15	14	13	10				

Table 3: Results for the Augerat instances with four different penalty settings

P5, while the distances are raised by only 30%. This difference can be explained by the fact that when distances rise, there is less buffer time to be allocated and the solutions are more prone to suffer overtime and delay penalties.

The right part of Table 3 shows the 'penalty ratio', which is the proportion

$$\sum_{R_r \in Z} \frac{\Theta(R_r)}{F(Z)}$$

of the total objective that corresponds to penalties. The average over all four experimental conditions is 9.0%. The lowest ratios are achieved for P5 and Prop, followed by P10, and the ratios for 1.3dist are by far the largest. We conclude that an increase in the distances has a substantial impact on the delay penalties.

4.3 VRP-SITW versus VRP

The addition of SITW to the VRP can be expected to affect the distance traveled and the number of vehicles used. To assess the effect, we compare the total distance in VRP-SITW with the optimal VRP solutions (taken from Ralphs (2010)). The details are provided in Table 4. For P5 and P10, the average distance increase is 3.3% and 3.7%, respectively, which shows that, at least as far as distance minimization is concerned, our heuristic solutions are rather close to optimal; the same observation can be made for Prop. For 1.3dist the VRP distances are scaled by a factor of 1.3, but this does not have an important impact on the distance increase. Overall, we conclude that the distance increase is not substantial for any of the settings.

4.4 VRP-SITW versus VRPTW

The goal of this section is to evaluate the benefits of the flexibility in setting time windows compared to exogenously predetermined time windows. To this aim, we work with 29 VRPTW instances from Solomon (1987). We compare the results of the VRP-SITW with the best-known solutions for the Solomon instances as reported in Solomon (2010).

Table 5 reports the results. For brevity we denote the travel time, which is equivalent to the distance, by T_F for the VRPTW (which has fixed time windows) and by T_S for the VRP-SITW. The number of vehicles required in the VRPTW is represented by K_F while the number of vehicles used by the VRP-SITW solution is written as K_S. The third column in Table 5 gives the ratio of the total travel times in both solutions. We observe that the VRP-SITW substantially reduces the travel time for instances with tight time windows such as those in the R1 and RC1 sets. Set C1, on the other hand, achieves zero penalty values, which can be read from the last column of the table. We conclude that these instances

Instance	Increase in distance			
	P5	P10	Prop	1.3dist
32 k5	101.1%	101.1%	102.7%	101.1%
33 k5	101.3%	102.7%	101.6%	101.3%
33 k6	101.2%	101.2%	100.7%	101.2%
34 k5	101.5%	101.4%	101.5%	101.4%
36 k5	102.5%	102.5%	101.3%	102.5%
37 k5	101.4%	101.4%	104.5%	101.4%
37 k6	102.0%	102.5%	102.1%	102.5%
38 k5	103.5%	103.5%	103.3%	103.5%
39 k5	102.6%	103.7%	100.9%	103.7%
39 k6	102.3%	102.3%	100.5%	104.5%
44 k6	102.4%	102.4%	105.6%	102.4%
45 k6	104.6%	104.6%	104.5%	103.6%
45 k7	103.1%	103.1%	101.9%	101.9%
46 k7	102.1%	102.1%	102.7%	102.1%
48 k7	103.9%	105.4%	105.0%	105.2%
53 k7	106.5%	106.5%	103.4%	106.5%
54 k7	102.0%	102.0%	103.6%	102.0%
55 k9	102.6%	102.6%	102.4%	102.6%
60 k9	102.7%	103.8%	103.4%	102.7%
61 k9	109.4%	109.4%	114.2%	109.4%
62 k8	105.0%	105.0%	104.0%	104.9%
63 k10	103.1%	109.5%	104.5%	109.5%
63 k9	103.4%	103.4%	104.0%	103.4%
64 k9	103.7%	103.7%	105.6%	103.7%
65 k9	106.1%	106.1%	104.8%	106.1%
69 k9	103.3%	103.3%	101.5%	103.3%
80 k10	104.4%	104.4%	106.0%	105.1%
Average	103.3%	103.7%	103.5%	103.6%

Table 4: Comparison of VRP-SITW with optimal VRP solutions for the Augerat instances

have quite unrestrictive time windows and exhibit a behavior similar to the VRP instances studied in Section 4.3. Across the datasets, the penalty component $\sum_{R_r \in Z} \Theta(R_r)$ comprises at most 6.3 % of the total objective value.

The fifth column of Table 5 displays the number of vehicles saved in VRP-SITW compared to VRPTW. A substantial reduction in the required number of vehicles is observed in the R1 and RC1 sets. In set C1, however, no such reduction is achieved. We conclude that those instances that allow for substantial reductions in travel times are eligible for similar improvements with respect to the number of vehicles.

Instance	T_F	T_S/T_F	K_F	$K_F - K_S$	$\sum_{R_r \in Z} \Theta(R_r)/F(Z)$
R101	1637.7	52.0%	20	12	6.3%
R102	1466.6	59.9%	18	10	4.5%
R103	1208.7	72.7%	14	6	4.5%
R104	971.5	89.7%	11	3	5.2%
R105	1355.3	63.5%	15	7	5.6%
R106	1251.98	68.7%	12	4	5.6%
R107	1064.6	80.8%	11	3	5.6%
R108	960.88	89.5%	9	1	5.6%
R109	1146.9	75.0%	13	5	5.6%
R110	1068	80.5%	12	4	5.6%
R111	1048.7	82.0%	12	4	5.6%
R112	982.14	87.6%	9	1	5.6%
C101	827.3	100.9%	10	0	0.0%
C102	827.3	100.9%	10	0	0.0%
C103	826.3	101.0%	10	0	0.0%
C104	822.9	101.4%	10	0	0.0%
C105	827.3	100.9%	10	0	0.0%
C106	827.3	100.9%	10	0	0.0%
C107	827.3	100.9%	10	0	0.0%
C108	827.3	100.9%	10	0	0.0%
C109	827.3	100.9%	10	0	0.0%
RC101	1619.8	61.9%	15	6	1.1%
RC102	1457.4	68.8%	14	5	1.1%
RC103	1258	79.4%	13	4	1.4%
RC104	1261.67	79.6%	11	2	1.7%
RC105	1513.7	66.2%	15	6	1.1%
RC106	1424.73	70.5%	11	2	1.7%
RC107	1207.8	83.2%	12	3	1.7%
Average		82.9%			2.7%

Table 5: Comparison of VRP-SITW with the best known VRPTW solutions for the Solomon instances

5. Conclusions

In this paper, we have analyzed the situation of carrier companies that face the problem of making routing decisions combined with the quotation of arrival times to their customers; we have referred to this setting by the term 'Self-Imposed Time Windows' (SITW). In the context of vehicle routing, the resulting VRP-SITW extends the VRP by the incorporation of customer-specific service aspects, reflected in the carrier company's ability to uphold the time windows once quoted, in a stochastic environment. In comparison with the VRP with exogenous time windows (VRPTW), the customer service requirement is somewhat relaxed, in that the service provider has *ex ante* flexibility in choosing a convenient time interval that will be quoted.

Our solution approach is a hybrid algorithm that comprises two main components: routing and scheduling. The routing component is handled via a tabu search procedure, while scheduling is performed by solving an LP model that implicitly inserts buffers into each route's schedule.

We have compared the VRP to VRP-SITW under different choices for penalty structures and distances. The results of our tests indicate that the VRP-SITW requires an average increase of some 3.5% in distance. Further research might focus on the impact of additional vehicles on the penalties.

Contrary to the VRP, the VRPTW exhibits substantial differences when compared to VRP-SITW. In most cases, the VRP-SITW requires significantly less distance and uses far less vehicles. Clearly, the VRP-SITW benefits greatly from its flexibility in setting the time windows. In our opinion, there is important potential in conducting an in-depth study of various flexibility levels in choosing delivery windows. Such a study can be beneficial, for instance, when negotiating service contracts. Another extension might look into the setting where only a subset of customers have fixed time windows. Furthermore, given some alterations the proposed model can also accommodate driving breaks, by using the buffers for the breaks. The proposed model establishes an *a priori* plan for a static environment. Yet another major extension of the model might incorporate the quotation of time windows for dynamically arriving orders. Finally, a trade-off may be conjectured between tardiness penalties and total travel times. Additional vehicles, for instance, will tend to improve the ability to uphold time windows but will generally increase travel times. Such trade-offs also offer opportunities for further work.

References

Augerat, P., J.M. Belenguer, E. Benavent, A. Corber, D. Naddef. 1998. Separating capacity constraints in the CVRP using tabu search. *European Journal of Operational Research* **106** 546–557.

Ballestín, F., R. Leus. 2008. Meta-heuristics for stable scheduling on a single machine. *Computers & Operations Research* **35** 2175–2192.

Bräysy, O., M. Gendreau. 2005a. Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Transportation Science* **39**(1) 104–118.

Bräysy, O., M. Gendreau. 2005b. Vehicle routing problem with time windows, part II: Metaheuristics. *Transportation Science* **39**(1) 119–139.

Daniels, R., J. Carrillo. 1997. β-robust scheduling for single-machine systems with uncertain processing times. *IIE Transactions* **29** 977–985.

Daniels, R., P. Kouvelis. 1995. Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science* **41** 363–376.

Finke, D.A., D.J. Medeiros, M.T. Traband. 2007. Multiple machine JIT scheduling: a tabu search approach. *International Journal of Production Research* **45**(21) 4899–4915.

Flisberg, P., B. Lidéna, M. Rönnqvist. 2009. A hybrid method based on linear programming and tabu search for routing of logging trucks. *Computers & Operations Research* **36**(4) 1122–1144.

Garcia, B.L., J.Y. Potvin, J.M. Rousseau. 1994. A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints. *Computers & Operations Research* **21**(9) 1025–1033.

Gendreau, M., A. Hertz, G. Laporte. 1994. A tabu search heuristic for the vehicle routing problem. *Management Science* **40**(10) 1276–1290.

Gendreau, M., G. Laporte, R. Séguin. 1996. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research* **44**(3) 469–477.

Gendreau, M, G. Laporte, M.M. Solomon. 1995. Single-vehicle routing and scheduling to minimize the number of delays. *Transportation Science* **29**(1) 56–62.

Herroelen, W., R. Leus. 2004. The construction of stable project baseline schedules. *European Journal of Operational Research* **156** 550–565.

Hertz, A., G. Laporte, M. Mittaz. 2000. A tabu search heuristic for the capacitated arc routing problem. *Operations Research* **48**(1) 129–135.

Hopp, W.J., M.L. Spearman. 1996. *Factory Physics, Foundations for Manufacturing Management*. Mc-Graw Hill.

Jabali, O., T. Van Woensel, A.G. de Kok, C. Lecluyse, H. Peremans. 2009. Time-dependent vehicle routing subject to time delay perturbations. *IIE Transactions* **41**(12) 1049–1066.

Kouvelis, P., R. Daniels, G. Vairaktarakis. 2000. Robust scheduling of a two-machine flow shop with uncertain processing times. *IIE Transactions* **32** 421–432.

Kouvelis, P., G. Yu. 1997. *Robust Discrete Optimization and its Applications*. Kluwer Academic Publishers.

Laporte, G. 1992. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research* **59**(3) 345–358.

Laporte, G., F. Louveaux, H. Mercure. 1992. The vehicle routing problem with stochastic travel times. *Transportation Science* **26**(3) 161–170.

Leus, R., W. Herroelen. 2005. The complexity of machine scheduling for stability with a single disrupted job. *Operations Research Letters* **33** 151–156.

Leus, R., W. Herroelen. 2007. Scheduling for stability in single-machine production systems. *Journal of Scheduling* **10** 223–235.

Li, X., P. Tian, S.C.H Leung. 2010. Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics* **125** 137–145.

Mitrović-Minć, S., G. Laporte. 2004. Waiting strategies for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B* **38** 635–655.

Or, I. 1976. *Traveling salesman-type combinatorial poroblems and their relation to the logistics of regional blood banking*. Ph.D. Thesis, Northwestern University, Evanston, Illinois.

Potvin, J.Y., J.M. Rousseau. 1995. An exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society* **46**(12) 1433–1446.

Puchinger, J., G.R. Raidl. 2005. Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. J. Mira, J.R. Álvarez, eds., *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach. Lecture Notes in Computer Science*, vol. 3562. Springer, 41–53.

Ralphs, T. 2010. Branch Cut and Price Resource Web. URL `http://www.branchandcut.org`. Last accessed October 2010.

Solomon, M.M. 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research* **35**(2) 254–265.

Solomon, M.M. 2010. VRPTW Benchmark Problems. URL `http://web.cba.neu.edu/~msolomon/problems.htm`. Last accessed April 2010.

Taillard, E., P. Badeau, M. Gendreau, F. Guertin, J.Y. Potvin. 1997. A tabu search heuristic for the vehicle routing problem with soft time windows. *Computers & Operations Research* **31**(2) 170–186.

Xiang, S., C. Chu, H. Chen. 2008. The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. *European Journal of Operational Research* **185**(2) 534–551.